Package 'mallet'

October 13, 2022

Type Package

Title An R Wrapper for the Java Mallet Topic Modeling Toolkit

Version 1.3.0

Date 2022-07-19

Maintainer Måns Magnusson <mons.magnusson@gmail.com>

Description An R interface for the Java Machine Learning for Language Toolkit (mallet) <http://mallet.cs.umass.edu/> to estimate probabilistic topic models, such as Latent Dirichlet Allocation. We can use the R package to read textual data into mallet from R objects, run the Java implementation of mallet directly in R, and extract results as R objects. The Mallet toolkit has many functions, this wrapper focuses on the topic modeling sub-package written by David Mimno. The package uses the rJava package to connect to a JVM.

License MIT + file LICENSE

URL https://github.com/mimno/RMallet

BugReports https://github.com/mimno/RMallet/issues

SystemRequirements java

Encoding UTF-8

Depends R (>= 3.6.3)

Imports rJava, checkmate

Suggests knitr, rmarkdown, dplyr, testthat

VignetteBuilder rmarkdown, knitr

RoxygenNote 7.2.0

LazyData TRUE

NeedsCompilation no

Author Måns Magnusson [cre, aut] (<https://orcid.org/0000-0002-0296-2719>), David Mimno [aut, cph] (<https://orcid.org/0000-0001-7510-9404>)

Repository CRAN

Date/Publication 2022-07-20 15:50:05 UTC

R topics documented:

`mallet-package` *An R Wrapper for the Java Mallet Topic Modeling Toolkit*

Description

An R interface for the Java Machine Learning for Language Toolkit (mallet) <http://mallet.cs.umass.edu/> to estimate probabilistic topic models, such as Latent Dirichlet Allocation. We can use the R package to read textual data into mallet from R objects, run the Java implementation of mallet directly in R, and extract results as R objects. The Mallet toolkit has many functions, this wrapper focuses on the topic modeling sub-package written by David Mimno. The package uses the rJava package to connect to a JVM.

References

The model, Latent Dirichlet allocation (LDA): *David M Blei, Andrew Ng, Michael Jordan. Latent Dirichlet Allocation. J. of Machine Learning Research, 2003.*

The Java toolkit: *Andrew Kachites McCallum. The Mallet Toolkit. 2002.*

Details of the fast sparse Gibbs sampling algorithm: *Limin Yao, David Mimno, Andrew McCallum. Streaming Inference for Latent Dirichlet Allocation. KDD, 2009.*

Hyperparameter optimization: *Hanna Wallach, David Mimno, Andrew McCallum. Rethinking LDA: Why Priors Matter. NIPS, 2010.*

`load.mallet.state` *Load a Mallet state into Mallet*

Description

This reads writes a current sampling state of mallet to file. The state contain hyperparameters α and β together with topic indicators.

Usage

```
load.mallet.state(topic.model, state.file)
```

Arguments

`topic.model` A `cc.mallet.topics.RTopicModel` object created by `MalletLDA`.

`state.file` File path to store the mallet state file to.

Value

a java `cc.mallet.topics.RTopicModel` object

`mallet.doc.topics` *Retrieve a matrix of topic weights for every document*

Description

This function returns a matrix with one row for every document and one column for every topic.

Usage

```
mallet.doc.topics(topic.model, normalized = FALSE, smoothed = FALSE)
```

Arguments

`topic.model` A `cc.mallet.topics.RTopicModel` object created by `MalletLDA`.

`normalized` If `TRUE`, normalize the rows so that each document sums to one. If `FALSE`, values will be integers (possibly plus the smoothing constant) representing the actual number of words of each topic in the documents.

`smoothed` If `TRUE`, add the smoothing parameter for the model (initial value specified as `alpha.sum` in `MalletLDA`). If `FALSE`, many values will be zero.

Value

a number of documents by number of topics matrix.

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[[”text”]],
                 stoplist = mallet_stoplist_file_path(”en”),
                 token.regexp = ”\\p{L}[\\p{L}\\p{P}]+\\p{L}”)

# Create topic model
topic.model <- MalletLDA(num.topics=10, alpha.sum = 1, beta = 0.1)
topic.model$loadDocuments(sotu.instances)

# Train topic model
topic.model$train(200)

# Extract results
doc_topics <- mallet.doc.topics(topic.model, smoothed=TRUE, normalized=TRUE)
topic_words <- mallet.topic.words(topic.model, smoothed=TRUE, normalized=TRUE)
top_words <- mallet.top.words(topic.model, word.weights = topic_words[2,], num.top.words = 5)

## End(Not run)
```

mallet.import *Import text documents into Mallet format*

Description

This function takes an array of document IDs and text files (as character strings) and converts them into a Mallet instance list.

Usage

```
mallet.import(
  id.array = NULL,
  text.array,
  stoplist = "",
  preserve.case = FALSE,
  token.regexp = ”[\\p{L}]+”
)
```

Arguments

`id.array` An array of document IDs. Default is `text.array` index.

`text.array` A character vector with each element containing a document.

`stoplist` The name of a file containing stopwords (words to ignore), one per line, or a character vector containing stop words. If the file is not in the current working directory, you may need to include a full path. Default is no stoplist.

`preserve.case` By default, the input text is converted to all lowercase.

`token.regexp` A quoted string representing a regular expression that defines a token. The default is one or more unicode letter: "[\\p{L}]+". Note that special characters must have double backslashes.

Value

a `cc/mallet/types/InstanceList` object.

See Also

`mallet.word.freqs` returns term and document frequencies, which may be useful in selecting stopwords.

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[["text"]],
                 stoplist = mallet_stoplist_file_path("en"),
                 token.regexp = "\\p{L}[\\p{L}\\p{P}]+\\p{L}")


## End(Not run)
```

`mallet.read.dir` *Import documents from a directory into Mallet format*

Description

This function takes a directory path as its only argument and returns a `data.frame` with two columns: <id> & <text>, which can be passed to the `mallet.import` function. This `data.frame` has as many rows as there are files in the `Dir`.

Usage

```
mallet.read.dir(Dir)
```

Arguments

`Dir` The path to a directory containing one document per file.

Value

a data.frame with file id and text content.

Note

This function was contributed to RMallet by Dan Bowen.

Author(s)

Dan Bowen

See Also

mallet.import

Examples

```
## Not run:
directory <- system.file("stoplists", package = "mallet")
stoplists <- mallet.read.dir(directory)

## End(Not run)
```

mallet.subset.topic.words

Estimate topic-word distributions from a sub-corpus

Description

This function returns a matrix of word probabilities for each topic similar to mallet.topic.words, but estimated from a subset of the documents in the corpus. The model assumes that topics are the same no matter where they are used, but we know this is often not the case. This function lets us test whether some words are used more or less than we expect in a particular set of documents.

Usage

```
mallet.subset.topic.words(
  topic.model,
  subset.docs,
  normalized = FALSE,
  smoothed = FALSE
)
```

Arguments

`topic.model` A `cc.mallet.topics.RTopicModel` object created by `MalletLDA`.

`subset.docs` A logical vector of `TRUE`/`FALSE` values specifying which documents should be used/included and which should be ignored.

`normalized` If `TRUE`, normalize the rows so that each topic sums to one. If `FALSE`, values will be integers (possibly plus the smoothing constant) representing the actual number of words of each type in the topics.

`smoothed` If `TRUE`, add the smoothing parameter for the model (initial value specified as `beta` in `MalletLDA`). If `FALSE`, many values will be zero.

Value

a number of topics by vocabulary size matrix for the the included documents.

See Also

`mallet.topic.words`

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[[”text”]],
                 stoplist = mallet_stoplist_file_path(”en”),
                 token.regexp = ”\\p{L}[\\p{L}\\p{P}]+\\p{L}”)

# Create topic model
topic.model <- MalletLDA(num.topics=10, alpha.sum = 1, beta = 0.1)
topic.model$loadDocuments(sotu.instances)

# Train topic model
topic.model$train(200)

# Extract subcorpus topic word matrix
post1975_topic_words <- mallet.subset.topic.words(topic.model, sotu[[”year”]] > 1975)
mallet.top.words(topic.model, word.weights = post1975_topic_words[2,], num.top.words = 5)

## End(Not run)
```

`mallet.top.words` *Get the most probable words and their probabilities for one topic*

Description

This function returns a data frame with two columns, one containing the most probable words as character values, the second containing the weight assigned to that word in the word weights vector you supplied.

Usage

```
mallet.top.words(topic.model, word.weights, num.top.words = 10)
```

Arguments

`topic.model` A `cc.mallet.topics.RTopicModel` object created by `MalletLDA`.

`word.weights` A vector of word weights for one topic, usually a row from the `topic.words` matrix from `mallet.topic.words`.

`num.top.words` The number of most probable words to return. If not specified, defaults to 10.

Value

a `data.frame` with the top terms (`term`) and their weights/probability (`weight`).

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[[”text”]],
                 stoplist = mallet_stoplist_file_path(”en”),
                 token.regexp = ”\\p{L}[\\p{L}\\p{P}]+\\p{L}”)

# Create topic model
topic.model <- MalletLDA(num.topics=10, alpha.sum = 1, beta = 0.1)
topic.model$loadDocuments(sotu.instances)

# Train topic model
topic.model$train(200)

# Extract top words
top_words <- mallet.top.words(topic.model, word.weights = topic_words[2,], num.top.words = 5)

## End(Not run)
```

mallet.topic.hclust *Return a hierarchical clustering of topics*

Description

Returns a hierarchical clustering of topics that can be plotted as a dendrogram. There are two ways of measuring topic similarity: topics may contain the some of the same words, or the may appear in some of the same documents. The balance parameter allows you to interpolate between the similarities determined by these two methods.

Usage

```
mallet.topic.hclust(
  doc.topics,
  topic.words,
  balance = 0.3,
  method = "euclidean",
  ...
)
```

Arguments

doc.topics	A documents by topics matrix of topic probabilities (see mallet.doc.topics).
topic.words	A topics by words matrix of word probabilities (see mallet.topic.words) .
balance	A value between 0.0 (use only document-level similarity) and 1.0 (use only word-level similarity).
method	method to use in dist to compute distance between topics. Defaults to euclidian.
...	Further arguments for hclust.

Value

An object of class hclust which describes the tree produced by the clustering process.

See Also

This function uses data matrices from mallet.doc.topics and mallet.topic.words using the hclust function.

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[["text"]],
                 stoplist = mallet_stoplist_file_path("en"),
                 token.regexp = "\\p{L}[\\p{L}\\p{P}]+\\p{L}")

# Create topic model
topic.model <- MalletLDA(num.topics=10, alpha.sum = 1, beta = 0.1)
topic.model$loadDocuments(sotu.instances)

# Train topic model
topic.model$train(200)

# Create hiearchical clusters of topics
doc_topics <- mallet.doc.topics(topic.model, smoothed=TRUE, normalized=TRUE)
topic_words <- mallet.topic.words(topic.model, smoothed=TRUE, normalized=TRUE)
topic_labels <- mallet.topic.labels(topic.model)
plot(mallet.topic.hclust(doc_topics, topic_words, balance = 0.3), labels=topic_labels)

## End(Not run)
```

`mallet.topic.labels` *Get strings containing the most probable words for each topic*

Description

This function returns a vector of strings, one for each topic, with the most probable words in that topic separated by spaces.

Usage

```
mallet.topic.labels(topic.model, topic.words = NULL, num.top.words = 3, ...)
```

Arguments

`topic.model`	A `cc.mallet.topics.RTopicModel` object created by `MalletLDA`.
`topic.words`	The matrix of topic-word weights returned by `mallet.topic.words` Default (NULL) is to use the `topic.model` to extract the `topic.words`.
`num.top.words`	The number of words to include for each topic. Defaults to 3.
`...`	Further arguments supplied to `mallet.topic.words`.

Value

a character vector with one element per topic

See Also

`mallet.topic.words` produces topic-word weights. `mallet.top.words` produces a data frame for a single topic.

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[["text"]],
                 stoplist = mallet_stoplist_file_path("en"),
                 token.regexp = "\\p{L}[\\p{L}\\p{P}]+\\p{L}")

# Create topic model
topic.model <- MalletLDA(num.topics=10, alpha.sum = 1, beta = 0.1)
topic.model$loadDocuments(sotu.instances)

# Train topic model
topic.model$train(200)

# Create hiearchical clusters of topics
doc_topics <- mallet.doc.topics(topic.model, smoothed=TRUE, normalized=TRUE)
topic_words <- mallet.topic.words(topic.model, smoothed=TRUE, normalized=TRUE)
topic_labels <- mallet.topic.labels(topic.model)
plot(mallet.topic.hclust(doc_topics, topic_words, balance = 0.3), labels=topic_labels)

## End(Not run)
```

mallet.topic.model.read

Load (read) and save (write) a topic from a file

Description

This function returns the topic model loaded from a file or stores a topic model to file.

Usage

```
mallet.topic.model.read(filename)

mallet.topic.model.load(filename)

mallet.topic.model.write(topic.model, filename)

mallet.topic.model.save(topic.model, filename)
```

Arguments

`filename`	The mallet topic model file
`topic.model`	A `cc.mallet.topics.RTopicModel` object created by `MalletLDA`.

`mallet.topic.words` *Retrieve a matrix of words weights for topics*

Description

This function returns a matrix with one row for every topic and one column for every word in the vocabulary.

Usage

```
mallet.topic.words(topic.model, normalized = FALSE, smoothed = FALSE)
```

Arguments

`topic.model` A `cc.mallet.topics.RTopicModel` object created by `MalletLDA`.

`normalized` If `TRUE`, normalize the rows so that each topic sums to one. If `FALSE`, values will be integers (possibly plus the smoothing constant) representing the actual number of words of each type in the topics.

`smoothed` If `TRUE`, add the smoothing parameter for the model (initial value specified as `beta` in `MalletLDA`). If `FALSE`, many values will be zero.

Value

a number of topics by vocabulary size matrix.

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[["text"]],
                 stoplist = mallet_stoplist_file_path("en"),
                 token.regexp = "\\p{L}[\\p{L}\\p{P}]+\\p{L}")

# Create topic model
topic.model <- MalletLDA(num.topics=10, alpha.sum = 1, beta = 0.1)
topic.model$loadDocuments(sotu.instances)

# Train topic model
topic.model$train(200)

# Extract results
doc_topics <- mallet.doc.topics(topic.model, smoothed=TRUE, normalized=TRUE)
topic_words <- mallet.topic.words(topic.model, smoothed=TRUE, normalized=TRUE)
top_words <- mallet.top.words(topic.model, word.weights = topic_words[2,], num.top.words = 5)
```

```
## End(Not run)
```

mallet.word.freqs *Descriptive statistics of word frequencies*

Description

This method returns a data frame with one row for each unique vocabulary word, and three columns: the word as a character value, the total number of tokens of that word type, and the total number of documents that contain that word at least once. This information can be useful in identifying candidate stopwords.

Usage

```
mallet.word.freqs(topic.model)
```

Arguments

topic.model	A cc.mallet.topics.RTopicModel object created by MalletLDA.

Value

a data.frame with the word type (word), the word frequency (word.freq), and the document frequency (doc.freq)

See Also

MalletLDA

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[["text"]],
                 stoplist = mallet_stoplist_file_path("en"),
                 token.regexp = "\\p{L}[\\p{L}\\p{P}]+\\p{L}")

# Create topic model
topic.model <- MalletLDA(num.topics=10, alpha.sum = 1, beta = 0.1)
topic.model$loadDocuments(sotu.instances)

# Get word frequencies
word_freqs <- mallet.word.freqs(topic.model)


## End(Not run)
```

MalletLDA *Create a Mallet topic model trainer*

Description

This function creates a java cc.mallet.topics.RTopicModel object that wraps a Mallet topic model trainer java object, cc.mallet.topics.ParallelTopicModel. Note that you can call any of the methods of this java object as properties. In the example below, I make a call directly to the `topic.model$setAlphaOptimization(20 50)` java method, which passes this update to the model itself.

Usage

```
MalletLDA(num.topics = 10, alpha.sum = 5, beta = 0.01)
```

Arguments

`num.topics` The number of topics to use. If not specified, this defaults to 10.

`alpha.sum` This is the magnitude of the Dirichlet prior over the topic distribution of a document. The default value is 5.0. With 10 topics, this setting leads to a Dirichlet with parameter $\alpha_k = 0.5$. You can intuitively think of this parameter as a number of "pseudo-words", divided evenly between all topics, that are present in every document no matter how the other words are allocated to topics. This is an initial value, which may be changed during training if hyperparameter optimization is active.

`beta` This is the per-word weight of the Dirichlet prior over topic-word distributions. The magnitude of the distribution (the sum over all words of this parameter) is determined by the number of words in the vocabulary. Again, this value may change due to hyperparameter optimization.

Value

a `cc.mallet.topics.RTopicModel` object

Examples

```
## Not run:
# Read in sotu example data
data(sotu)
sotu.instances <-
   mallet.import(id.array = row.names(sotu),
                 text.array = sotu[["text"]],
                 stoplist = mallet_stoplist_file_path("en"),
                 token.regexp = "\\p{L}[\\p{L}\\p{P}]+\\p{L}")

# Create topic model
topic.model <- MalletLDA(num.topics=10, alpha.sum = 1, beta = 0.1)
topic.model$loadDocuments(sotu.instances)
```

```
# Train topic model
topic.model$train(200)

# Extract results
doc_topics <- mallet.doc.topics(topic.model, smoothed=TRUE, normalized=TRUE)
topic_words <- mallet.topic.words(topic.model, smoothed=TRUE, normalized=TRUE)
top_words <- mallet.top.words(topic.model, word.weights = topic_words[2,], num.top.words = 5)

## End(Not run)
```

mallet_jar *Return the mallet jar filename(s)*

Description

Return the mallet jar filename(s)

Usage

```
mallet_jar(full.names = FALSE)

mallet.jar(full.names = FALSE)
```

Arguments

full.names	a logical value. If TRUE, the directory path is prepended to the file names to give a relative file path. If FALSE, the file name(s) (rather than paths) are returned.

Details

Mallet is implemented as a jar-file in the mallet R package. This function returns the file name and file path for that file(s)

mallet_stoplist_file_path *Return the file path to the mallet stoplists*

Description

Return the file path to the mallet stoplists

Usage

```
mallet_stoplist_file_path(language = "en")

mallet.stoplist.file.path(language = "en")
```

Arguments

`language`	language to return stoplist for. Defaults to engligs ([en]).

Details

Returns the path to the mallet stop word list. See [mallet_supported_stoplists()] for which stoplists that are included.

`mallet_supported_stoplists` *Mallet supported stoplists*

Description

Mallet supported stoplists

Usage

```
mallet_supported_stoplists()

mallet.supported.stoplists()
```

Details

return vector with included stoplists

`save.mallet.instances` *Load and save mallet instances from/to file*

Description

This function returns the topic model loaded from a file.

Usage

```
save.mallet.instances(instances, filename)

load.mallet.instances(filename)
```

Arguments

`instances`	An `cc/mallet/types/InstanceList instanceList` object to save/write to file.
`filename`	The filename to save to or load from.

`save.mallet.state` *Save a Mallet state to file*

Description

This function writes a current sampling state of mallet to file. The state contain hyperparameters α and β together with topic indicators.

The state file can be read into R using the function

Usage

```
save.mallet.state(topic.model, state.file)
```

Arguments

`topic.model`	A `cc.mallet.topics.RTopicModel` object created by `MalletLDA`.
`state.file`	File path (.gz format) to store the mallet state file to.

`sotu` *State of the Union Adresses.*

Description

A dataset containing State of the Union Adresses by paragraph from 1946 to 2000.

Usage

```
sotu
```

Format

A `tibble` `data.frame` with 6816 rows and 3 variables:

year Year of the adress.

paragraph The paragraph of the address.

text The address content.

Source

https://en.wikipedia.org/wiki/State_of_the_Union

Index